Econ 603 - A1 - Javier Fernandez

Assignment #1 ; Class: Econ 613 ; Name: Javier Fernandez ; Date: March 1st, 2021 ;

Econ 613 - Assignment 1 - Javier Fernandez

Preliminaries —–

```
# Libraries
library(tidyverse)
# install.packages("matlib")
library(matlib)
```

—– PART 1 ——

0. Load Data —–

```
dat_student <- read.csv("C:/Users/javie/OneDrive/Documents/Duke - MAE/Academic/Spring 2021/ECON 613 - Ap
dat_school <- read.csv("C:/Users/javie/OneDrive/Documents/Duke - MAE/Academic/Spring 2021/ECON 613 - App
dat_position <- read.csv("C:/Users/javie/OneDrive/Documents/Duke - MAE/Academic/Spring 2021/ECON 613 - A
```

1. Missing data —–

For easier calculations, I am gathering information to show one choice per row (six rows per student)

```
programs <- dat_student %>% select(1:4,11:18) %>%
  pivot_longer(cols=c("choicepgm1","choicepgm2","choicepgm3","choicepgm4",
                      "choicepgm5","choicepgm6"),names_to="Choice_num",
               names_prefix = "choicepgm",
               values_to="Program")

schools <- dat_student %>% select(1:10,17:18) %>%
  pivot_longer(cols=c("schoolcode1","schoolcode2","schoolcode3","schoolcode4",
                      "schoolcode5","schoolcode6"),names_to="Choice_num",
               names_prefix = "schoolcode",
               values_to="School")
dat_student_clean <- merge(programs,schools)
```

a. Number of students

```
nrow(dat_student) # or max(dat_student$X)
```

```
## [1] 340823
```

```
# answer: the data contains 340,823 distinct students
```

b. Number of schools

```
dat_student_clean %>% group_by(School) %>% summarise(Count=n()) %>%
  filter(School!="" | !is.na(School)) %>% nrow()
```

```
## [1] 640
```

```
# answer: the data contains 640 different schools
```

c. Number of programs

```
dat_student_clean %>% group_by(Program) %>% summarise(Count=n()) %>%
  filter(Program!="" | !is.na(Program)) %>% nrow()
```

```
## [1] 33
```

```
# answer: the data contains 33 distinct programs
```

d. Number of choices

```
# Number of distinct choices:
dat_student_clean %>% group_by(School,Program) %>% summarise(Count=n()) %>%
  filter(Program!="" | School!="") %>% nrow()
```

```
## [1] 3085
```

```
# answer: There are 3,085 distinct choices: combinations of schools and programs
```

e. Missing test scores

```
sum(is.na(dat_student$score))
```

```
## [1] 179887
```

```
# answer: There are 179,887 students missing test scores
```

f. Apply to the same school

```
dat_student_clean %>% group_by(X) %>% filter(Program!="") %>%
    summarise(Number_of_schools=n_distinct(School)) %>% filter(Number_of_schools==1) %>% nrow()
```

```
## [1] 663
```

```
# answer: 663 students applied to the same school in all their cases, irrespective to the
#         number of programs they applied to.
```

g. Apply to less than 6 choices

```
dat_student_clean %>% group_by(X) %>% filter(Program=="") %>%
  summarise(Number_of_Programs=n())  %>%  nrow()
```

```
## [1] 20988
```

```
# answer: 20,988 students applied to less than 6 choices.
```

2. Data ——

```
# To do this we first have to filter the student data by only keeping the
# the student was admitted to.

program_admitted <-   dat_student_clean %>% filter(Choice_num==rankplace)

admission_stats <- program_admitted %>% group_by(School,Program) %>%
  summarize(cutoff=min(score),quality=mean(score),size=n())

### erasing duplicates in dat_school
dat_school_clean <- dat_school[!duplicated(dat_school$schoolcode),]

choice_lvl_data <- left_join(admission_stats,dat_school_clean[,-1],by= c("School"="schoolcode"))

head(choice_lvl_data)
```

```
## # A tibble: 6 x 9
## # Groups:   School [1]
##   School Program  cutoff quality  size schoolname     sssdistrict ssslong ssslat
##    <int> <chr>     <int>   <dbl> <int> <chr>          <chr>         <dbl>  <dbl>
## 1  10101 Agricul~    288    310.    49 EBENEZER SENI~ Accra Metr~  -0.197   5.61
## 2  10101 Business    305    325.   100 EBENEZER SENI~ Accra Metr~  -0.197   5.61
## 3  10101 General~    316    330.   100 EBENEZER SENI~ Accra Metr~  -0.197   5.61
## 4  10101 General~    299    329.    50 EBENEZER SENI~ Accra Metr~  -0.197   5.61
## 5  10101 Home Ec~    284    301.    49 EBENEZER SENI~ Accra Metr~  -0.197   5.61
## 6  10101 Visual ~    296    312.    50 EBENEZER SENI~ Accra Metr~  -0.197   5.61
```

3. Distance ——

```
# Getting the coordinates for the junior high school
program_admitted_location <- left_join(program_admitted,dat_position[,-1],by= c("jssdistrict"="jssdistri

program_admitted_location <- left_join(program_admitted_location[,-1],choice_lvl_data,
                                       by=c("School"="School","Program"="Program"))

# Renaming variables for simplicity
program_admitted_location <- program_admitted_location %>%
                              rename(jsslong=point_x,jsslat=point_y,
                                     sss_code=School)

# Constructing distance variable
program_admitted_location <- program_admitted_location %>%
                    mutate(distance=sqrt((69.172*(ssslong-jsslong)*cos(jsslat/57.3))^2 +
                                           (69.172*(ssslat-jsslat))^2))

head(program_admitted_location)
```

```
##   score agey male                           jssdistrict rankplace Choice_num
## 1   249   16    0                          Agona Swedru         5          5
## 2   254   19    1 Abura/Asebu/Kwamankese (Abura Dunkwa)         2          2
## 3   277   17    0 Abura/Asebu/Kwamankese (Abura Dunkwa)         4          4
## 4   236   16    0 Abura/Asebu/Kwamankese (Abura Dunkwa)         3          3
## 5   237   18    1        Ajumako/Enyan/Essiam (Ajumako)         1          1
## 6   262   16    0            Twifo Hemang (Twifo Praso)         6          6
##          Program sss_code    jsslong   jsslat cutoff  quality size
## 1   General Arts    30403 -0.7552425 5.617353    208 245.2105   38
## 2    Agriculture    30403 -1.1970884 5.130001    219 241.9333   15
## 3 Home Economics    30403 -1.1970884 5.130001    215 248.3750    8
## 4   General Arts    30403 -1.1970884 5.130001    208 245.2105   38
## 5   General Arts    30403 -1.0053846 5.401725    208 245.2105   38
## 6    Agriculture    30403 -1.5597034 5.572999    219 241.9333   15
##                        schoolname                           sssdistrict
## 1 ABAKRAMPA SENIOR HIGH TECHNICAL Abura/Asebu/Kwamankese (Abura Dunkwa)
## 2 ABAKRAMPA SENIOR HIGH TECHNICAL Abura/Asebu/Kwamankese (Abura Dunkwa)
## 3 ABAKRAMPA SENIOR HIGH TECHNICAL Abura/Asebu/Kwamankese (Abura Dunkwa)
## 4 ABAKRAMPA SENIOR HIGH TECHNICAL Abura/Asebu/Kwamankese (Abura Dunkwa)
## 5 ABAKRAMPA SENIOR HIGH TECHNICAL Abura/Asebu/Kwamankese (Abura Dunkwa)
## 6 ABAKRAMPA SENIOR HIGH TECHNICAL Abura/Asebu/Kwamankese (Abura Dunkwa)
##     ssslong   ssslat distance
## 1 -1.197088 5.130001 45.40499
## 2 -1.197088 5.130001  0.00000
## 3 -1.197088 5.130001  0.00000
## 4 -1.197088 5.130001  0.00000
## 5 -1.197088 5.130001 22.96873
## 6 -1.197088 5.130001 39.52487
```

4. Descriptive Characteristics —

```r
# Total sample
program_admitted_location %>% group_by(Choice_num) %>%
  summarise(Mean_cutoff=mean(cutoff),
            Stdev_cutoff=sd(cutoff),
            Mean_quality=mean(quality),
            Stdev_quality=sd(quality),
            Mean_distance=mean(distance,na.rm = TRUE),
            Stdev_distance=sd(distance,na.rm = TRUE))
```

```
## # A tibble: 6 x 7
##   Choice_num Mean_cutoff Stdev_cutoff Mean_quality Stdev_quality Mean_distance
##   <chr>            <dbl>        <dbl>        <dbl>         <dbl>         <dbl>
## 1 1                 285.        59.7          311.          53.0          35.2
## 2 2                 278.        51.4          304.          44.7          33.9
## 3 3                 263.        44.0          290.          37.5          28.4
## 4 4                 249.        38.1          278.          31.9          22.7
## 5 5                 210.         8.19         252.          12.9          31.8
## 6 6                 210.         8.58         249.          11.2          31.2
## # ... with 1 more variable: Stdev_distance <dbl>
```

By School

```r
program_admitted_location %>% group_by(schoolname) %>%
                          summarise(Mean_cutoff=mean(cutoff),
                          Stdev_cutoff=sd(cutoff),
                          Mean_quality=mean(quality),
                          Stdev_quality=sd(quality),
                          Mean_distance=mean(distance,na.rm = TRUE),
                          Stdev_distance=sd(distance,na.rm = TRUE)) %>% head()
```

```
## # A tibble: 6 x 7
##   schoolname Mean_cutoff Stdev_cutoff Mean_quality Stdev_quality Mean_distance
##   <chr>            <dbl>        <dbl>        <dbl>         <dbl>         <dbl>
## 1 ABAKRAMPA~        212.         5.48         244.          2.85          22.7
## 2 ABETIFI P~        267.         9.92         297.          6.53          45.3
## 3 ABETIFI T~        208.         9.11         247.          7.15          13.0
## 4 ABOR SENI~        210.         3.47         245.          2.36          27.7
## 5 ABUAKWA S~        324.         9.70         343.          8.29          36.9
## 6 ABURAMAN ~        204.         8.00         250.          4.78          22.3
## # ... with 1 more variable: Stdev_distance <dbl>
```

By Program

```r
program_admitted_location %>% group_by(Program) %>%
  summarise(Mean_cutoff=mean(cutoff),
            Stdev_cutoff=sd(cutoff),
```

```
           Mean_quality=mean(quality),
           Stdev_quality=sd(quality),
           Mean_distance=mean(distance,na.rm = TRUE),
           Stdev_distance=sd(distance,na.rm = TRUE)) %>%  head()
```

```
## # A tibble: 6 x 7
##   Program Mean_cutoff Stdev_cutoff Mean_quality Stdev_quality Mean_distance
##   <chr>         <dbl>        <dbl>        <dbl>         <dbl>         <dbl>
## 1 Accoun~        206.        13.9          245.         10.9           21.9
## 2 Agric.~        213.         5.09         256.          3.46          55.8
## 3 Agricu~        246.        39.3          275.         34.5           27.8
## 4 Auto B~        298          0            320.          3.01          32.3
## 5 Block ~        217.        14.4          258.         16.3           30.0
## 6 Busine~        268.        52.2          298.         44.2           30.7
## # ... with 1 more variable: Stdev_distance <dbl>
```

Differentiated by quantiles (This is to be interpreted as the mean cutoff of the schools quantile x students will go to.)

```
program_admitted_location <- program_admitted_location %>% mutate(quantile=case_when(
  score<quantile(score,0.1)~1,
  score>=quantile(score,0.1) & score<quantile(score,0.2)~2,
  score>=quantile(score,0.2) & score<quantile(score,0.3)~3,
  score>=quantile(score,0.3) & score<quantile(score,0.4)~4,
  score>=quantile(score,0.4) & score<quantile(score,0.5)~5,
  score>=quantile(score,0.5) & score<quantile(score,0.6)~6,
  score>=quantile(score,0.6) & score<quantile(score,0.7)~7,
  score>=quantile(score,0.7) & score<quantile(score,0.8)~8,
  score>=quantile(score,0.8) & score<quantile(score,0.9)~9,
  score>=quantile(score,0.9) ~10,
))

program_admitted_location %>% group_by(quantile) %>%  summarise(Mean_cutoff=mean(cutoff),
                                        Stdev_cutoff=sd(cutoff),
                                        Mean_quality=mean(quality),
                                        Stdev_quality=sd(quality),
                                        Mean_distance=mean(distance,na.rm = TRUE),
                                        Stdev_distance=sd(distance,na.rm = TRUE))
```

```
## # A tibble: 10 x 7
##    quantile Mean_cutoff Stdev_cutoff Mean_quality Stdev_quality Mean_distance
##       <dbl>       <dbl>        <dbl>        <dbl>         <dbl>         <dbl>
##  1        1        210.         9.19         246.          10.7          25.1
##  2        2        220.        14.2          254.          11.5          26.0
##  3        3        229.        17.9          261.          13.1          26.9
##  4        4        239.        21.3          269.          15.3          27.7
##  5        5        251.        24.3          279.          18.1          29.6
##  6        6        265.        26.4          292.          20.4          30.8
##  7        7        278.        27.7          304.          21.8          31.3
##  8        8        295.        30.2          319.          24.0          32.9
##  9        9        324.        29.7          345.          24.7          35.5
```

```
## 10       10        366.        29.5          385.          27.0          43.7
## # ... with 1 more variable: Stdev_distance <dbl>
```

By rank choice and by quantile

```
program_admitted_location %>% group_by(quantile) %>%  summarise(Mean_cutoff=mean(cutoff),
                                        Stdev_cutoff=sd(cutoff),
                                        Mean_quality=mean(quality),
                                        Stdev_quality=sd(quality),
                                        Mean_distance=mean(distance,na.rm = TRUE),
                                        Stdev_distance=sd(distance,na.rm = TRUE))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 10 x 7
##    quantile Mean_cutoff Stdev_cutoff Mean_quality Stdev_quality Mean_distance
##       <dbl>       <dbl>        <dbl>        <dbl>         <dbl>         <dbl>
##  1        1        210.         9.19         246.          10.7          25.1
##  2        2        220.        14.2          254.          11.5          26.0
##  3        3        229.        17.9          261.          13.1          26.9
##  4        4        239.        21.3          269.          15.3          27.7
##  5        5        251.        24.3          279.          18.1          29.6
##  6        6        265.        26.4          292.          20.4          30.8
##  7        7        278.        27.7          304.          21.8          31.3
##  8        8        295.        30.2          319.          24.0          32.9
##  9        9        324.        29.7          345.          24.7          35.5
## 10       10        366.        29.5          385.          27.0          43.7
## # ... with 1 more variable: Stdev_distance <dbl>
```

—— PART 2 ——

5. Data creation ——

```
set.seed(123)

# X1
X1 <- runif(10000,min=1,max=3)

# X2
X2 <- rgamma(10000,shape = 3,scale = 2)

# X3
X3 <- rbinom(10000,size=1,prob = 0.3)

# Error term
error <- rnorm(10000,mean=2,sd=1)

# Create Y and Ydum
```

```r
# Y
par <- c(0.5,1.2,-0.9,0.1)
Y <- par[1] + par[2]*X1 + par[3]*X2 + par[4]*X3 + error

# Ydum
Ydum <- as.numeric(Y>mean(Y))
```

6. OLS —–

```r
# Correlation Y and X1. How different is it from 1.2?
# answer: the result is 0.21.
#Being Y a linear function of X1 we would have expected the correlation to be larger.

cor(Y,X1)
```

```
## [1] 0.216015
```

```r
# Calculate the coefficients

regressors <- as.matrix(t(rbind(rep(1,10000),X1,X2,X3)),ncol=4)

betas <-inv(t(regressors)%*%regressors)%*%(t(regressors)%*%Y)
# betas are the coefficients for the OLS estimation

resids <- Y-regressors%*%betas
sigma_2 <- as.numeric(t(resids)%*%resids/(10000-4))
var_cov_matrix <- sigma_2*inv(t(regressors)%*%regressors)
std_errors <- sqrt(diag(var_cov_matrix))
# std_errors are the standard errors of the coefficients
coef_stderrors <- cbind(betas,std_errors)
colnames(coef_stderrors) <- c("Coefs","Std. Errors")

print(coef_stderrors) # Answer
```

```
##            Coefs Std. Errors
## [1,]  2.49051092 0.040620582
## [2,]  1.19777741 0.017358659
## [3,] -0.89640329 0.002875798
## [4,]  0.08781299 0.021694686
```

7. Discrete choice —–

```r
  # The linear probability model can be estimated by OLS
  # Function:
    linear_prob_model <- function(Y,regressors){
        betas <-inv(t(regressors)%*%regressors)%*%(t(regressors)%*%Y)
        resids <- Y-regressors%*%betas
```

```
        sigma_2 <- as.numeric(t(resids)%*%resids/(nrow(regressors)-ncol(regressors)))
        var_cov_matrix <- sigma_2*inv(t(regressors)%*%regressors)
        std_errors <- sqrt(diag(var_cov_matrix))
        coef_stderrors <- cbind(betas,std_errors)
        colnames(coef_stderrors) <- c("Coefs","Std. Errors")
        return(coef_stderrors) # Answer
    }

  # Estimation
regressors <- as.matrix(t(rbind(rep(1,10000),X1,X2,X3)),ncol=4)

# Results of the linear probability model (Coefs and regressors)
results_lpm <- linear_prob_model(Ydum,regressors)


####   7.b  Probit ----

      # Probit function
      probit_likelihood = function(coefs,x1,x2,x3,y)
      {
        xbeta           = coefs[1] + coefs[2]*x1 + coefs[3]*x2 + coefs[4]*x3
        pr              = pnorm(xbeta)
        pr[pr>0.999999] = 0.999999
        pr[pr<0.000001] = 0.000001
        like            = y*log(pr) + (1-y)*log(1-pr)
        return(-sum(like))
      }

### Estimation

    # Result Probit----
start = runif(4)
res_probit = optim(start,fn=probit_likelihood,method="BFGS",control=list(trace=6,REPORT=10,maxit=10000)
```

7.a. Linear probability model —-

```
## initial  value 24689.277344
## iter  10 value 2214.628584
## final  value 2213.313307
## converged
```

```
fisher_info_probit = solve(res_probit$hessian)
prop_sigma_probit  = sqrt(diag(fisher_info_probit))


####   7.c  Logit ----

      # Logit function
      logit_likelihood = function(y,x1,x2,x3,coefs)
      {
        xbeta           = coefs[1] + coefs[2]*x1 + coefs[3]*x2 + coefs[4]*x3
        pr              = exp(xbeta)/(1+exp(xbeta))
        pr[pr>0.999999] = 0.999999
```

```
        pr[pr<0.000001] = 0.000001
        like           = y*log(pr) + (1-y)*log(1-pr)
        return(-sum(like))
      }

### Estimation

    # Result Logit ----
start = runif(4)
res_logit = optim(start,fn=logit_likelihood,method="BFGS",control=list(trace=6,REPORT=10,maxit=10000),x
```

```
## initial  value 20564.984166
## iter  10 value 2224.622518
## final  value 2223.017344
## converged
```

```
fisher_info_logit = solve(res_logit$hessian)
prop_sigma_logit  = sqrt(diag(fisher_info_logit))
```

```
  # Final Results ----
results = cbind(par,results_lpm[,1],results_lpm[,2],
                res_probit$par,prop_sigma_probit,res_logit$par,prop_sigma_logit)
colnames(results) = c("True parameter","LPM: est","LPM :se","Probit: est","Probit: :se",
                      "Logit: est","Logit: :se")
results
```

```
##      True parameter     LPM: est      LPM :se Probit: est Probit: :se
## [1,]            0.5  0.885860391 0.0136557488  3.04275799  0.10007791
## [2,]            1.2  0.146150735 0.0058356006  1.17235964  0.04292123
## [3,]           -0.9 -0.102941654 0.0009667803 -0.90546589  0.01858996
## [4,]            0.1 -0.008099353 0.0072932778 -0.01124976  0.04647615
##       Logit: est Logit: :se
## [1,]  5.42655537 0.18557806
## [2,]  2.10059552 0.07936241
## [3,] -1.61851052 0.03670961
## [4,] -0.01963215 0.08323293
```

```
  # Answer
# 1. The LPM, which is the only comparable model in terms of coefficients, produces estimates
# really different from the true parameters. At least they are all un the correct direction.
# Using a t-test with 95% confidence, the intercept, X1 and X2 are significant.
# This is not the case for X3.
Significance_lpm <- abs(results_lpm[,1]/results_lpm[,2])>1.96
Significance_lpm
```

```
## [1]  TRUE  TRUE  TRUE FALSE
```

```
# 2. The Probit model coefficients are not directly comparable with the true parameters.
# We can observe that the sign of the estimates is correct for all but X3.
# Using a t-test with 95% confidence, the intercept, X1 and X2 are significant.
# This is not the case for X3.
Significance_probit <-abs(res_probit$par/prop_sigma_probit)>1.96
Significance_probit
```

```
## [1]  TRUE  TRUE  TRUE FALSE
```

```
# 3. The Logit model coefficients are not directly comparable with the true parameters.
# We can observe that the sign of the estimates is correct for all but X3.
# Using a t-test with 95% confidence, the intercept, X1 and X2 are significant.
# This is not the case for X3.
Significance_logit <- abs(res_logit$par/prop_sigma_logit)>1.96
Significance_logit
```

```
## [1]  TRUE  TRUE  TRUE FALSE
```

8. Marginal effects ——

Average Marginal effects

```
#### Probit average marginal effects
Xbeta_probit <- regressors %*% as.matrix(res_probit$par)
mgl_effects_probit <- dnorm(Xbeta_probit)%*% t(as.matrix(res_probit$par))

mean_mgleff_probit <- colMeans(mgl_effects_probit)

#### Logit average marginal effects
Xbeta_logit <- regressors %*% as.matrix(res_logit$par)
mgl_effects_logit <- (plogis(Xbeta_logit)*(1-plogis(Xbeta_logit)))%*%t(as.matrix(res_logit$par))

mean_mgleff_logit <- colMeans(mgl_effects_logit)
```

Marginal effects at the mean

```
#### Probit
mean_Xbeta_probit <- mean(regressors) %*% t(as.matrix(res_probit$par))
mgl_eff_mean_probit <- as.numeric(dnorm(mean_Xbeta_probit)*res_probit$par)
mgl_eff_mean_probit
```

```
## [1]  2.040878e-11  1.176231e-02 -4.014827e-02 -4.486481e-03
```

```
#### Logit
mean_Xbeta_logit <- mean(regressors) %*% t(as.matrix(res_logit$par))
mgl_eff_mean_logit <- as.numeric((plogis(mean_Xbeta_logit)*(1-plogis(mean_Xbeta_logit)))*t(as.matrix(res
mgl_eff_mean_logit
```

```
## [1]  1.899787e-05  1.598576e-02 -3.644384e-02 -4.905504e-03
```

Bootstraping to compute the marginal errors

```r
# Probit and logit model

set.seed(322)     # Setting seed
iter    = 199     # Number of iterations
n_obs = length(X1)  # Number of observations
number_var = length(res_probit$par) # Number of parameters (including intercept)

# To save values
outs_probit = mat.or.vec(iter,number_var)
outs_logit = mat.or.vec(iter,number_var)
outs_mgleff_probit = mat.or.vec(iter,number_var)
outs_mgleff_logit = mat.or.vec(iter,number_var)

# Bootstrap
for (i in 1:iter){
  # New sample
  new_sample = sample(1:n_obs,n_obs,rep=TRUE)
  dat_sample = as.data.frame(regressors[new_sample,])

  # Models
  probit_res = glm(data = dat_sample,Ydum ~ X1 + X2 + X3, family = binomial(link = "probit"))
  logit_res = glm(data = dat_sample,Ydum ~ X1 + X2 + X3, family = binomial(link = "logit"))

  # Saving coefficients
  outs_probit[i,] = coef(probit_res)
  outs_logit[i,] = coef(logit_res)

  # Mean marginal effects
  outs_mgleff_probit[i,] = colMeans(dnorm(Xbeta_probit)%*% t(as.matrix(outs_probit[i,])))
  outs_mgleff_logit[i,] = colMeans((plogis(Xbeta_logit)*(1-plogis(Xbeta_logit)))%*%t(as.matrix(outs_logi
  }

# Compute standard errors
sd_mgleff_probit = apply(outs_mgleff_probit,2,sd)
sd_mgleff_logit = apply(outs_mgleff_logit,2,sd)


## Answers

mgleffects <- cbind(mean_mgleff_probit,mgl_eff_mean_probit,sd_mgleff_probit,
                    mean_mgleff_logit,mgl_eff_mean_logit,sd_mgleff_logit)
mgleffects <- mgleffects[-1,]
colnames(mgleffects) <- c("Probit: Avg Mgl Eff","Probit: Mgl Eff at Mean",
                          "Probit: SE of Mgl Eff","Logit: Avg Mgl Eff",
                          "Logit: Mgl Eff at Mean","Logit: SE of Mgl Eff")

rownames(mgleffects) <- c("X1","X2","X3")
mgleffects # Answer
```

```
##    Probit: Avg Mgl Eff Probit: Mgl Eff at Mean Probit: SE of Mgl Eff
## X1         0.143808297             0.011762305          0.0026012905
## X2        -0.111069593            -0.040148270          0.0004519666
## X3        -0.001379959            -0.004486481          0.0033842010
```

```
##    Logit: Avg Mgl Eff Logit: Mgl Eff at Mean Logit: SE of Mgl Eff
## X1         0.14403075            0.015985763         0.0023284781
## X2        -0.11097581           -0.036443837         0.0004045455
## X3        -0.00134611           -0.004905504         0.0030290037
```

Answers

1. Probit model: The average marginal effect for X1 is 0.1438, the only positive one. This means that a unit change in X1 leads to an increase of 0.1438 percentage points in the probablity of having Ydum=1. X2 and X3 have lower marginal effects and in the opposite direction. In the case of the marginal effects at the mean, all the effects are smaller, indicating that probably this measure is not representative of the data. Finally, the standard errors of the marginal effects are relatively small for X1 and X3, but are large for X3. This indicates uncertainty on the estimation of X3.

2. Logit model: The average marginal effect for X1 is 0.144, the only positive one. This means that a unit change in X1 leads to an increase of 0.144 percentage points in the probablity of having Ydum=1. X2 and X3 have lower marginal effects and in the opposite direction. These results are similar to those of the probit model. In the case of the marginal effects at the mean, all the effects are smaller, indicating that probably this measure is not representative of the data. Finally, the standard errors of the marginal effects are relatively small for X1 and X3, but are large for X3. This indicates uncertainty on the estimation of X3.